Weighted network estimation by the use of topological graph metrics

Weighted network estimation by the use of topological graph metrics

Loukianos Spyrou and Javier Escudero

Abstract—Topological metrics of graphs provide a natural way to describe the prominent features of various types of networks. Graph metrics describe the structure and interplay of graph edges and have found applications in many scientific fields. In this work, graph metrics are used in network estimation by developing optimisation methods that incorporate prior knowledge of a network's topology. The derivatives of graph metrics are used in gradient descent schemes for weighted undirected network denoising, network completion , and network decomposition. The successful performance of our methodology is shown in a number of toy examples and real-world datasets. Most notably, our work establishes a new link between graph theory, network science and optimisation.

Index Terms—graph metric derivatives, graph theory, network completion, network denoising, network decomposition, optimisation

1 Introduction

Graph theory has found applications in many scientific fields in an attempt to analyse interconnections between phenomena, measurements, and systems. It provides a data structure that naturally express those interconnections and also provides a framework for further analysis [1], [2]. A graph consists of a set of nodes and edges describing the connections between the nodes. The edges of binary graphs take the values of either 1 or 0 indicating the presence or absence of a connection, while in weighted graphs the edges are described by weights indicating the strength of the connection. Graphs have been extensively used in a variety of applications in network science such as biological networks, brain networks, and social networks [3]–[6].

Graph metrics are functions of a graph's edges and characterize one or several aspects of network connectivity [7]–[9]. Local metrics deal with the relation of specific nodes to the network structure while global metrics describe properties of the whole network. Graph metrics are largely used to describe the functional integration or segregation of a network, quantify the centrality of individual regions, detect community structure, characterize patterns of interconnections, and test resilience of networks to abrupt changes.

Estimation of a network's structure or properties has been performed in a variety of contexts. Although a single definition does not exist, network estimation can include any algorithm or method that detects, enhances, generates, or increases some quality measure of networks. Link prediction and network completion deal with predicting the existence of missing edges based on the observed links in a binary graph [10]–[14]. Also, in [13] the prediction of missing nodes was attempted. So far such methods have dealt with detecting only whether a link (edge) exists or not, not with the estimation of its weight. Typical applications include predicting the appearance of future connections in social [10], [15] or biological [16] networks. The network reconstruction problem deals with composing networks that satisfy specific properties [17]–[20]. This can be particularly useful when building null models for hypothesis testing. Network inference problem attempts to identify a network structure where the edges have been corrupted by a diffusion process through the network [21], [22]. More recently, reducing the noise in social networks has been attempted in [23]–[25].

In this work, we assume that prior information is available regarding an observed weighted undirected network. This prior information comes in the form of estimates of a number of topological graph metrics of the network. We utilise these estimates in an optimisation framework in order to adjust the weights of the observed network to satisfy those properties. There are many real-world cases where there is knowledge of a network's structure but not exact or reliable network information [17], e.g. strength of connections between banks is known but exact connections are hidden for privacy issues in bank networks [26], modularity metrics of brain networks are similar between subjects [27], properties of the constituent networks may be known for mixed networks [24]. We demonstrate the utility of our methodology in three schemes.

Firstly, a network denoising scheme, where an observed network is a noisy version of an underlying noise-free network. By considering the error between the observed network's graph metrics and the true network's known metrics, in an iterative gradient descent process on the network's weights, the resulting network is closer to the noise-free one. Using the node degrees as priors has been performed in binary networks in [24] and in terms of network reconstruction the knowledge of degrees has been employed in weighted networks in [17], [28]. In [23], the transitivity of a network was used in a network diffusion scheme to change a social network's weights but without a specific stopping point. In [25], denoising has been attempted in the context of removing weak ties between users in a social networks. Here, we provide analytical and empirical proofs on the utility of denoising schemes that are based on the optimisation of various graph metrics through gradient descent.

Secondly, we develop a weighted network completion

This work was supported by EPSRC, UK, Grant No. EP/N014421/1
Loukianos Spyrou and Javier Escudero are with the School of Engineering, University of Edinburgh, EH9 3FB, U.K.

scheme where some of the weights of the network's edges are missing. Similarly to the two previous schemes, we adapt the missing weights such that the whole network obtains specific values for known graph metrics. Weighted network completion has not been performed in the literature per se, only the closely related matrix completion problem [29] and matrix completion on graphs [30] where the completion is aided by assuming that the rows and columns of the matrix form communities.

Finally, we develop a network decomposition scheme for the cases where an observed network is an additive mixture of two networks. Assuming that graph metrics are known for the two constituent networks we derive an algorithm that can estimate the networks by enforcing them to have specific values for graph metrics while keeping the reconstruction error between the original and estimated mixture small. Network decomposition has been traditionally applied in a different context, on decomposing graphs with disjoint nodes [31]. For factored binary graphs untangling has been performed by taking into account the degree distribution of the constituent graphs [24]. Here, we not only consider the degrees of a network and decomposing it into subgraphs (i.e. multiple graphs with disjoint nodes) but we use multiple graph metrics in additive graph mixtures.

Therefore, in this paper, we provide a comprehensive description of theoretical and empirical results on the use of optimisation of graph metrics for various problems in network estimation. In section 2 we give some basic definitions and a brief introduction to graph theory and graph metrics. The network estimation methods are shown in Section 3 with the details of the three schemes, denoising 3.1, completion 3.2, and decomposition 3.3. In section 3.4 we derive the graph metrics derivatives that are used in the optimisation methods. In section 4 we apply our methodology to a number of toy examples and real data and in section 5 we put the results into context and discuss the utility that our method provides. Section 6 concludes the paper.

2 Graph metrics

A weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ is defined by a finite set of nodes (or vertices) $\mathcal{V}$ with $|\mathcal{V}| = n$, a set of edges $\mathcal{E}$ of the form $(v_i, v_j) \in \mathcal{E}$ with $|\mathcal{E}| = n^2 - n$ and a weighted adjacency matrix $\mathbf{W}$ with $w_{ii} = 0 \,\forall\, i$. In this work, we consider undirected graphs for which $\mathbf{W}$ is symmetric, i.e. $w_{ij} = w_{ji}$ or $\mathbf{W} = \mathbf{W^T}$. The entries w_{ij} in the weighted adjacency matrix $\mathbf{W}$ (weight matrix from now on) indicate the strength of connection between nodes. We assume that networks are normalised, i.e. $w_{ij} \in [0, 1]$.

Graph metrics are scalar functions of the weight matrix, i.e. $f(\mathbf{W}) : \mathbb{R}^{n^2} \to \mathbb{R}$. Global metrics map the weight matrix into a single value and therefore attempt to simply quantify a specific property of a network. Local metrics on the other hand, quantify some property of the network separately for each node $i \in \{1...n\}$ with $f_i(\mathbf{W}) \in \mathbb{R}^{n^2} \to \mathbb{R}$, potentially resulting in n functions and n separate values.

Although graph metrics were originally defined on binary (unweighted) networks, the conversion to weighted metrics is usually but not always straightforward [8], [32]–[36]. The main motivation of this study can be recognised by pointing out that a network can be adjusted by changing the matrix $\mathbf{W}$ so that it obtains a certain value for some graph measure. There are numerous graph metrics that describe various features of networks [8], [9]. The main properties that they measure are:

- Integration (ability of the network to combine information from distributed nodes).
- Segregation (ability of the network for specialised processing in densely interconnected node groups).
- Centrality (characteristic that describes the presence of regions responsible for integrating total activity).
- Resilience (ability of a network to withstand weight changes).
- Motifs (presence of specific network patterns).
- Community structure (separation of a network to various functional groups).
- Small-worldness (highly segregated *and* integrated nodes).

There are graph metrics that contain quantities which are themselves a product of an optimisation procedure (e.g. module structure, shortest path length). These quantities are considered outside of the scope of this work.

2.1 Definitions

Here we show some definitions, notations and useful matrices that are used in the following sections.

n	:	number of nodes
m	:	number of graph metrics
$\mathbf{S}_{ij} = \{1_{ij}\}, \{0_{\neg i \| \neg j}\}$	:	matrix of zeros except at (i, j)
$\{\mathbf{A}\}_{ij} = a_{ij} = tr\{\mathbf{A}\mathbf{S}_{ji}\}$	:	$(i, j)_{th}$ element of matrix $\mathbf{A}$
$tr\{\mathbf{A}\} = \sum_{i=1}^{n} \{\mathbf{A}\}_{ii}$	:	sum of diagonal elements
$\mathbf{1}_n$	:	column vector of ones
$\mathbf{O}_n = \mathbf{1}_n \mathbf{1}_n^T$	:	matrix of ones
$\mathbf{H}_n = \mathbf{O}_n - \mathbf{I}_n$	:	all ones except at diagonal
$\mathbf{R}_j = \{1_{ij} \forall i\}, \{0_{\neg ij \forall i}\}$	:	matrix with ones at column j
$\sum_{ij} a_{ij} = tr\{\mathbf{A}\mathbf{O}_n\}$	:	sum of all matrix entries
$\mathbf{A} \circ \mathbf{B}$	:	Hadamard (element-wise) product

3 Network Estimation

In this section we formulate the optimisation methodologies for the three schemes considered in this work.

3.1 Denoising

Suppose that a weight matrix $\mathbf{W}$ of a network is corrupted by additive noise:

$$\mathbf{W}_e = \mathbf{W} + \mathbf{E} \tag{1}$$

The error matrix $\mathbf{E}$ can be considered as a network unrelated to the network structure being considered. For example, $\mathbf{E}$ could be social calls when trying to detect suspicious calls in social networks [24], the effect of volume conduction in EEG based brain network connectivity, or measurement noise. A different type of noise that occurs in networks, the effect of missing values is treated in section 3.2.

If we assume that we have estimates of M differentiable graph metrics of the original $\mathbf{W}$, i.e. $f_m(\mathbf{W}) = K_m$ where $m \in \{1, ..., M\}$, then we can formulate a cost function that measures the deviation of the observed weight matrix's metrics $f_m(\mathbf{W_e})$ to the estimates K_m as:

$$c(\mathbf{W}_e) = \sum_m e_m^2(\mathbf{W}_e) = \sum_m \left(f_m(\mathbf{W}_e) - K_m\right)^2 \tag{2}$$

The error is minimised with gradient descent updates on $\mathbf{W_e}$:

$$\mathbf{W}_e^{(t+1)} = \mathbf{W}_e^t - \mu \sum_m e_m(\mathbf{W}_e^t) \frac{df_m(\mathbf{W}_e^t)}{d\mathbf{W}_e^t} \tag{3}$$

where t is the iteration index and μ the learning rate. For the case of $m = 1$, Equation (3) describes the traditional single function gradient descent. For $m > 1$, it can be considered an equally weighted sum method of multiobjective optimisation, motivated by the fact that the graph metrics are in the same range due to normalisation. Multiobjective optimisation enables the weighting of different metrics to accommodate priorities on which are more important for a specific task. Such weighting is considered above the scope of this work. The full denoising procedure is described in Algorithm 1. Note that values below 0 and above 1 are truncated to zero and one respectively since we assume that the networks are normalised.

In Appendix A we provide a proof that for convex cost functions $c(\mathbf{W})$, denoising guarantees error reduction. The implication of that is that when a graph metric results in a convex cost function, such as the degrees (k_i^w) of the network, then the optimisation of Algorithm 1 with $f(\mathbf{W}) = \frac{1}{n}\sum_i k_i^w$ will always converge to a solution $\hat{\mathbf{W}}$ that is closer to the original network $\mathbf{W}$ than $\mathbf{W}_e$ is. For non convex metrics, there is no such guarantee. In Section 3.1 we show empirical results on the extent of that effect. In the Appendix C we show the proof that cost functions based on graph metrics such as the degree are convex. In general, linear functions of the weights of the adjacency matrix are convex whereas nonlinear functions of the weights are not.

3.2 Completion

For the case that a set $\mathcal{M}$ of entries of the weight matrix are missing, we can perform matrix completion by:

$$\mathbf{W}_{ic}^{(t+1)} = \mathbf{W}_{ic}^t - \mu \sum_m e_m(\mathbf{W}_{ic}^t) \left(\frac{df_m(\mathbf{W}_{ic}^t)}{d\mathbf{W}_{ic}^t} \circ \mathbf{S}_{\mathcal{M}} \right) \tag{4}$$

Algorithm 1 Denoising of corrupted network $\mathbf{W}_e$ through the use of graph measure estimates K_m

OUTPUT: $\hat{\mathbf{W}}$
INPUTS: $\mathbf{W_e}$, K_m
Initialise $t = 0, \mathbf{W}^0 = \mathbf{W}_e$,
E=$\sum_m e_m^2(\mathbf{W}^0) = \sum_m \left(f_m(\mathbf{W}^0) - K_m\right)^2$
while $E > \epsilon$ **do**
  $\mathbf{W}^{(t+1)} = \mathbf{W}^t - \mu \sum_m e_m(\mathbf{W}^t) \frac{df_m(\mathbf{W}^t)}{d\mathbf{W}^t}$
  if $w_{ij} < 0$ then $w_{ij} = 0$, $w_{ij} > 1$ then $w_{ij} = 1$
  E=$\sum_m e_m^2(\mathbf{W}^{t+1}) = \sum_m \left(f_m(\mathbf{W}^{t+1}) - K_m\right)^2$
  $t = t + 1$
end while
Denoised network: $\hat{\mathbf{W}} = \mathbf{W}^t$

where $\mathbf{S}_{\mathcal{M}}$ is a matrix with ones at the set of missing entries and zeroes everywhere else. Similar to the previous two cases, the assumption is that if the true graph metrics are known, gradient descent will adjust the missing weights close to their true values. The missing entries of the incomplete weight matrix $\mathbf{W}_{ic}$ can be initialised to the most likely value of the network (w). This can be considered as a denoising procedure with the missing weights equal to 'noisy' weights of value w. In Algorithm 2 we describe the network completion procedure.

Algorithm 2 Completing the missing entries of network $\mathbf{W}_{ic}$ through the use of graph measure estimates K_m

OUTPUT: $\hat{\mathbf{W}}_c$
INPUTS: $\mathbf{W}_{ic}$, K_m, set of missing entries $\mathcal{M}$
Initialise $t = 0, \{\mathbf{W}_{ic}^0\}_{ij} = w$ with $(i, j) \in \mathcal{M}$,
E=$\sum_m e_m^2(\mathbf{W}_{ic}^0) = \sum_m \left(f_m(\mathbf{W}_{ic}^0) - K_m\right)^2$
while $E > \epsilon$ **do**
  $\mathbf{W}_{ic}^{(t+1)} = \mathbf{W}_{ic}^t - \mu \sum_m e_m(\mathbf{W}_{ic}^t) \frac{df_m(\mathbf{W}_{ic}^t)}{d\mathbf{W}_{ic}^t} \circ \mathbf{S}_{\mathcal{M}}$
  if $w_{ij} < 0$ then $w_{ij} = 0$, $w_{ij} > 1$ then $w_{ij} = 1$
  E=$\sum_m e_m^2(\mathbf{W}_{ic}^{t+1}) = \sum_m \left(f_m(\mathbf{W}_{ic}^{t+1}) - K_m\right)^2$
  $t = t + 1$
end while
Complete network estimate: $\hat{\mathbf{W}}_c = \mathbf{W}_{ic}^t$

3.3 Decomposition

Suppose that we observe a mixed network that arises as a combination of two networks:

$$\mathbf{W}_f = \mathbf{W}_1 + \mathbf{W}_2 \tag{5}$$

If we have estimates of some topological properties of the two networks, i.e. $\{f_m^1(\mathbf{W}_1) = K_m^1\}, \{f_m^2(\mathbf{W}_2) = K_m^2\}$, then we can utilise these information to infer the networks from their mixture. This could be accomplished separately for each network using Algorithm 1. However, since we know the mixture $\mathbf{W}_f$, we utilise that in the following optimisation problem:

$$\begin{aligned} \underset{\mathbf{W}_1, \mathbf{W}_2}{\text{argmin}} \quad & \sum_m \left(f_m^1(\mathbf{W}_1) - K_m^1\right)^2 + \left(f_m^2(\mathbf{W}_2) - K_m^2\right)^2 \\ \text{subject to} \quad & ||\mathbf{W}_f - (\mathbf{W}_1 + \mathbf{W}_2)||_{\text{F}}^2 \leq \xi \end{aligned}$$

We solve this optimisation problem with alternating minimisation since it is a function of two matrices. We fix one of the two weight matrices and solve the following optimisation problem for the other one in an alternating fashion:

$$\underset{\mathbf{W}_1}{\operatorname{argmin}} \quad \sum_m \left(f_m^1(\mathbf{W}_1) - K_m^1\right)^2 + \lambda||\mathbf{W}_1 - (\mathbf{W}_f - \mathbf{W}_2)||_{\mathrm{F}}^2 \tag{6}$$

$$\underset{\mathbf{W}_2}{\operatorname{argmin}} \quad \sum_m \left(f_m^2(\mathbf{W}_2) - K_m^2\right)^2 + \lambda||\mathbf{W}_2 - (\mathbf{W}_f - \mathbf{W}_1)||_{\mathrm{F}}^2 \tag{7}$$

where the constraint has been incorporated into the cost function through the penalty parameter λ. Each separate minimisation, Algorithm 3, resembles Algorithm 1 but in this case deviations from $\mathbf{W}_f - \mathbf{W}_1$ or $\mathbf{W}_f - \mathbf{W}_2$ are penalised. This whole procedure is shown in Algorithm 4.

Algorithm 3 Constrained network optimisation based on graph metrics K_m and a constraint that penalises deviations from a reference network $\mathbf{Y}$. The penalty is adjustable through the parameter λ.

OUTPUT: $\hat{\mathbf{W}}$
INPUTS: $\mathbf{Y}$, K_m, λ
Initialise $t = 0$, $\mathbf{W}^0 = \mathbf{Y}$
E=$\sum_m e_m^2(\mathbf{W}^0) = \sum_m \left(f_m(\mathbf{W}^0) - K_m\right)^2$
while $E > \epsilon$ **do**
  $\mathbf{W}^{(t+1)} = \mathbf{W}^t - \mu\left(\sum_m e_m(\mathbf{W}^t)\frac{df_m(\mathbf{W}^t)}{d\mathbf{W}^t} + \lambda(\mathbf{W} - \mathbf{Y})||\mathbf{W} - \mathbf{Y}||_{\mathrm{F}}^{\mathbf{2}}\right)$
  if $w_{ij} < 0$ then $w_{ij} = 0$, $w_{ij} > 1$ then $w_{ij} = 1$
  E=$\sum_m e_m^2(\mathbf{W}^{t+1}) = \sum_m \left(f_m(\mathbf{W}^{t+1}) - K_m\right)^2$
  $t = t + 1$
end while
Constrained estimate: $\hat{\mathbf{W}} = \mathbf{W}^t$

Algorithm 4 Alternating minimisation procedure for network decomposition of a mixed network $\mathbf{W}_f$ through known graph metrics for the individual networks K_m^1, K_m^2.

OUTPUT: $\hat{\mathbf{W}}_1, \hat{\mathbf{W}}_2$
INPUTS: $\mathbf{W}_f$, K_m^1, K_m^2, λ
Initialise $t = 0$,
$\mathbf{W}_1^0 = \text{Algorithm1}\{\mathbf{W}_f, K_m^1\}$
$\mathbf{W}_2^0 = \text{Algorithm1}\{\mathbf{W}_f, K_m^2\}$
$R = ||\mathbf{W_m} - (\mathbf{W}_1^0 + \mathbf{W}_2^0)||_{\mathrm{F}}^2$
while $R > \epsilon$ **do**
  $\mathbf{W}_1^{t+1} = \text{Algorithm3}\{\mathbf{W}_f - \mathbf{W}_2^t, \quad K_m^1, \lambda\}$
  if $w_{ij}^1 < 0$ then $w_{ij}^1 = 0$, $w_{ij}^1 > 1$ then $w_{ij}^1 = 1$
  $\mathbf{W}_2^{t+1} = \text{Algorithm3}\{\mathbf{W}_f - \mathbf{W}_1^{t+1}, K_m^2, \lambda\}$
  if $w_{ij}^2 < 0$ then $w_{ij}^2 = 0$, $w_{ij}^2 > 1$ then $w_{ij}^2 = 1$
  $R = ||\mathbf{W_f} - (\mathbf{W}_1^{t+1} + \mathbf{W}_2^{t+1})||_{\mathrm{F}}^2$
  $t = t + 1$
end while
Optimised estimates: $\mathbf{W}_1, \mathbf{W}_2$

The motivation for the procedure in Algorithm 4 is the following. Consider estimating the two weight matrices only by using the denoising algorithm for each one separately. The estimate of $\mathbf{W}_1$ can be written as $\hat{\mathbf{W}}_1 = \mathbf{W}_1 + \mathbf{E}_1$ where $\mathbf{E}_1$ indicates the error from the true weight matrix $\mathbf{W}_1$. Similarly for $\mathbf{W}_2$ resulting in $\hat{\mathbf{W}}_f = \mathbf{W}_1 + \mathbf{W}_2 + \mathbf{E}_1 + \mathbf{E}_2 = \mathbf{W}_f + \mathbf{E}$. Therefore, it is evident that the estimates of the weight matrices are not ideal whenever $||\mathbf{E}||_F > 0$. Note that if $||\mathbf{E}||_F = 0$ it does not necessarily imply that the estimates of the weight matrices are optimal since it is possible that $\mathbf{E}_1 = -\mathbf{E}_2$. It would be optimal if we could constraint the estimate e.g. $\hat{\mathbf{W}}_1$ as:

$$\begin{aligned} &\underset{\mathbf{W}_1}{\operatorname{argmin}} && \sum_m \left(f_m^1(\mathbf{W}_1) - K_m^1\right)^2 \\ &\text{subject to} && ||\mathbf{E}_1||_{\mathrm{F}}^2 = 0 \end{aligned}$$

However that would require knowledge of the true weight matrix $\mathbf{W}_1$. Instead, note that:
$\hat{\mathbf{W}}_1 - (\mathbf{W}_f - \hat{\mathbf{W}}_2) = \mathbf{W}_1 + \mathbf{E}_1 - (\mathbf{W}_1 + \mathbf{W}_2 - \mathbf{W}_2 - \mathbf{E}_2) = \mathbf{E}_1 + \mathbf{E}_2$.
Therefore by reducing $||\mathbf{W}_1 - (\mathbf{W}_f - \mathbf{W}_2)||_F$ in Eq. (6), we are reducing the total error. In other words we are solving the following constrained optimisation problem:

$$\begin{aligned} &\underset{\mathbf{W}_1}{\operatorname{argmin}} && \sum_m \left(f_m^1(\mathbf{W}_1) - K_m^1\right)^2 \\ &\text{subject to} && ||\mathbf{E}_1 + \mathbf{E}_2||_{\mathrm{F}}^2 \leq \xi \end{aligned} \tag{8}$$

The inequality is incorporated such that $||\mathbf{E}_2||_F \geq 0$ when optimising $\mathbf{W}_1$ and vice versa. Consider the extreme cases. Firstly, when $\xi = 0$. That would imply that $\lambda = \infty$ which would render the graph measure optimisation ineffective. On the other hand, a large ξ implies small λ which would render the constraint ineffective. In our implementation we adjust the λ parameter such that the reconstruction error, i.e. $\mathbf{W}_f - \hat{\mathbf{W}}_f$ slowly decreases over the iterations of the alternating minimisation algorithm. Note the well known fact that there is a one-to-one correspondence between ξ in Eq. 8 and λ in Eq. 6.

3.4 Derivatives of graph metrics

In this section we derive the expressions for the derivatives of popular graph metrics that describe some important properties of networks. We deal with: degree, average neighbour degree, transitivity, clustering coefficient, modularity. More details can be found in Appendix B.

3.4.1 Degree

The degree of a node i describes the connection strength of that node to all other nodes:

$$k_i^w = \sum_j w_{ij} = tr\{\mathbf{W}\mathbf{R}_i\} \tag{9}$$

with the degree derivative being:

$$\frac{\partial k_i^w}{\partial \mathbf{W}} = \mathbf{R}_i^T \tag{10}$$

Since $\mathbf{R}_i$ is non-zero only for column i it can be computed efficiently as:

$$\frac{\partial k_i^w}{\partial \mathbf{w}_i} = \mathbf{1}_n{}^T \tag{11}$$

where $\mathbf{w}_i$ is the i_{th} column of $\mathbf{W}$.

3.4.2 Average neighbour degree - resilience

The average neighbour degree for node i is given by:

$$ND_i = \frac{\sum_j w_{ij}k_j^w}{k_i^w} = \frac{tr\{\mathbf{W}^2\mathbf{R}_i\}}{tr\{\mathbf{W}\mathbf{R}_i\}} = \frac{\rho}{\tau} \tag{12}$$

The derivative of the average neighbour degree is:

$$\frac{\partial ND_i}{\partial \mathbf{W}} = \frac{\tau(\mathbf{W}\mathbf{R}_i + \mathbf{R}_i\mathbf{W})^T - \rho\mathbf{R}_i^T}{\tau^2} \tag{13}$$

3.4.3 Transitivity - segregation

The transitivity is a global measure of the segregation of a network and here we defined it as (see also [23]):

$$T = \frac{\sum_{ijh} w_{ij}w_{ih}w_{jh}}{\sum_{ij}\sum_h w_{ih}w_{jh}} = \frac{tr\{\mathbf{W^3}\}}{tr\{\mathbf{W}\mathbf{H}_n\mathbf{W}\}} = \frac{\alpha}{\beta} \tag{14}$$

The transitivity derivative is:

$$\frac{\partial T}{\partial \mathbf{W}} = \left(\frac{3\beta\mathbf{W}^2 - \alpha(\mathbf{W}\mathbf{H}_n + \mathbf{H}_n\mathbf{W})}{\beta^2}\right) \tag{15}$$

3.4.4 Clustering coefficient - segregation

The clustering coefficient for node i is a local measure of the clustering of a network. It is defined as:

$$C_i = \frac{\sum_{jh} w_{ij}w_{ih}w_{jh}}{\sum_{jh} w_{ij}w_{ih}} = \frac{\{\mathbf{W}^3\}_{ii}}{\{\mathbf{W}\mathbf{H_n}\mathbf{W}\}_{ii}} = \frac{tr\{\mathbf{S}_{ii}\mathbf{W}^3\}}{tr\{\mathbf{S}_{ii}\mathbf{W}\mathbf{H}_n\mathbf{W}\}} = \frac{\gamma_i}{\zeta_i} \tag{16}$$

The derivative of the clustering coefficient is:

$$\frac{\partial C_i}{\partial \mathbf{W}} = \left(\frac{3\zeta_i\sum_r^2\left(\mathbf{W}^r\mathbf{S}_{ii}\mathbf{W}^{2-r}\right) - \gamma_i(\mathbf{S}_{ii}^T\mathbf{W}^T\mathbf{H}^T + \mathbf{H}^T\mathbf{W}^T\mathbf{S}_{ii}^T)}{\zeta_i^2}\right)$$

3.4.5 Modularity - community structure

Modularity metrics the tendency of a network to be divided into modules. Here we deal with optimising the modularity in terms of the network weights, not in terms grouping nodes into modules. Modularity can be written as (see also [37]):

$$M = \frac{1}{l^w}\sum_{ij}\left(w_{ij} - \frac{k_i^w k_j^w}{l^w}\right)\delta_{ij} \tag{17}$$

where $\delta_{ij} = 1$ whenever nodes i and j belong to the same module and zero otherwise.

The modularity derivative is expressed as:

$$\frac{\partial M}{\partial \mathbf{W}} = \frac{\partial m_1}{\partial \mathbf{W}} - \frac{\partial m_2}{\partial \mathbf{W}} \tag{18}$$

with:

$$\frac{\partial m_1}{\partial \mathbf{W}} = \frac{l^w\mathbf{\Delta} - \theta\mathbf{O_n^T}}{(l^w)^2} \tag{19}$$

and:

$$\frac{\partial m_2}{\partial \mathbf{W}} = \sum_{r=1}^{n}\frac{l^w(\mathbf{C_r}\mathbf{W}\mathbf{\Delta^T} + \mathbf{C_r^T}\mathbf{W}\mathbf{\Delta}) - 2\xi_r\mathbf{O_n^T}}{(l^w)^3} \tag{20}$$

where $\mathbf{C_r}$ is a circular shift matrix that shifts down the rows of the matrix on the right by $r-1$ and $\xi_r = \mathbf{W^T}\mathbf{C_r}\mathbf{W}\mathbf{\Delta^T}$. See Appendix B.B for more details.

3.4.6 Local and global metrics

Any graph measure f_i that operates locally on node i can be cast into its global (full network) form by evaluating the gradient as the average of the nodes' derivatives:

$$\frac{\partial f}{\partial \mathbf{W}} = \frac{1}{n}\sum_i \frac{\partial f_i}{\partial \mathbf{W}} \tag{21}$$

4 Results

4.1 Denoising

4.1.1 Synthetic Networks

In this section we show results of applying the denoising algorithm for various cases. We create synthetic undirected networks of three types:

- Random complete network , $w_{ij} \sim \mathcal{U}[0,1]$
- Scale free weighted network where the degrees are distributed with the power law and the non zero edges are given weights $w_{ij} \sim \mathcal{U}[0,1]$. The network was created according to [38] with average degree of 5.
- A modular network where the weights exhibit community structure in a number of modules. The network was created with the BCT toolbox [8]. The network consists of 8 modules and 90% of the non-zero weights in the modules.

For each case we add a noise matrix where each entry of $\mathbf{E}$, $e_{ij} \sim \mathcal{N}(0,1)$ is normally distributed with mean 0 and standard deviation 1:

$$\mathbf{W}_e = \mathbf{W} + \sigma\mathbf{E} \tag{22}$$

Weights of $\mathbf{W}_e$ that go below 0 are set to zero, and subsequently the weight matrix is normalised by dividing by its maximum value. In that way we guarantee that all elements of $\mathbf{W}_e$ are between 0 and 1.

Firstly, we show the error reduction of the scheme for various noise levels and networks of 128 nodes. We define error reduction as the ratio of the error of the denoised network $\hat{\mathbf{W}}$ to the error of the noisy network $\mathbf{W}_e$:

$$er = 1 - \frac{||\hat{\mathbf{W}} - \mathbf{W}||_{\mathrm{F}}}{||\mathbf{W}_e - \mathbf{W}||_{\mathrm{F}}} \tag{23}$$

Error reduction is measure in the domain $(-\infty, 1]$ where 1 indicates perfect denoising. Negative values indicate larger error after applying the denoising algorithm.

In Figures 1, 2 and 3 we show the error reduction for an increasing noise level and different graph metrics for the random, scale-free and modular network respectively. Each noise level considers the average of 50 noise matrix realisations. Note that even though the error reduction increases as the noise increases, in absolute terms the error always increases.

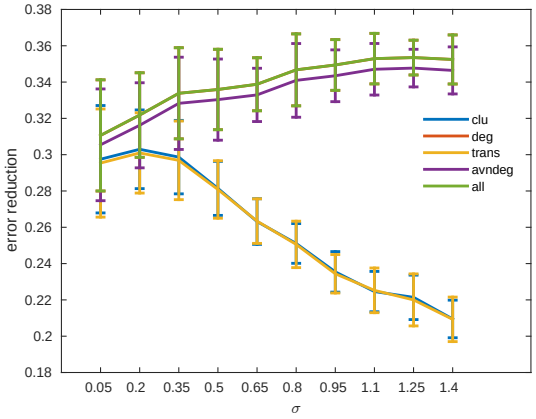

Fig. 1. Denoising of a random complete network with 128 nodes. We show the error reduction for an increasing noise level and various graph metrics. Note that the degree coincides with the combination of graph metrics in this figure. This indicates that for the cases that the degree is known no other network metrics are necessary.

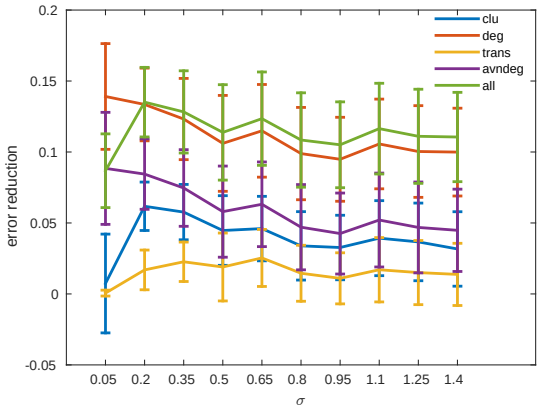

Fig. 2. Denoising of a scale-free network with 128 nodes. We show the error reduction for an increasing noise level and various graph metrics. Clustering based measures suffer in this type of network due to the decreased magnitude of those measures in scale-free networks.

Next, we show the error reduction in terms of the number of nodes in the network and a noise level of $\sigma = .5$, see Figure 4.

4.1.2 Real EEG data

The denoising algorithm was applied on two electroencephalography (EEG) datasets on a memory task from Alzheimer's patients and control subjects [39]. In dataset-1, there were 128-channel recordings from 13 patients with

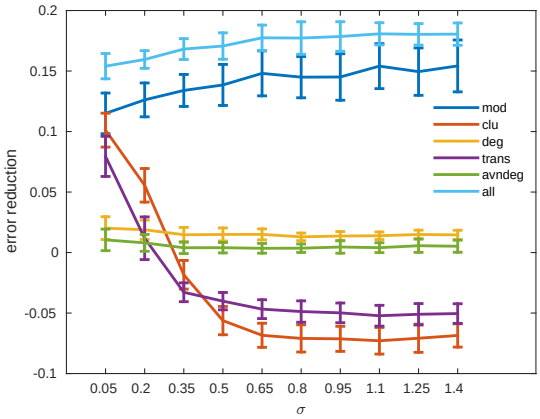

Fig. 3. Denoising of a modular network with 128 nodes and 8 modules and 90% of the weights in the modules. We show the error reduction for an increasing noise level and various graph metrics. Note that for the transitivity and clustering coefficient the estimated network had a larger error than without applying the denoising algorithm. This can be explained by noting that the weights are clustered in modules whereas the denoising algorithm is free to adapt any weight.

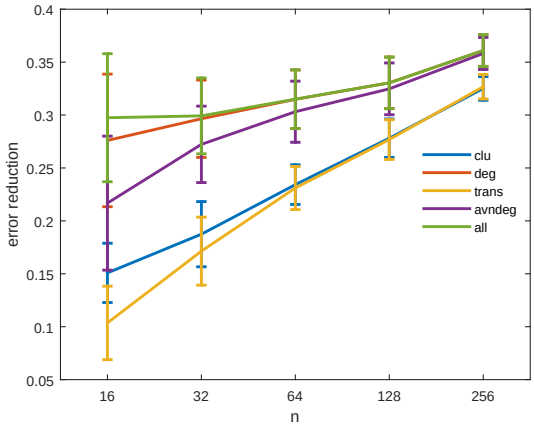

Fig. 4. Effect of the number of nodes on the error reduction and a constant $\sigma = .5$. Although the error reduction increases with the number of nodes there is an increase in absolute error as well.

mild cognitive impairment (MCI) and 19 control subjects while for dataset-2 there were 64-channel recordings from 10 patients with familial Alzheimer's disease (FAD) and 10 control subjects. For both datasets we selected the common subset of 28 electrodes that have the exact same locations on the scalp. For each subject we split the recording into 1 second epochs and computed the connectivity network matrix of the first 50 epochs and of each epoch separately. We used the imaginary part of coherence as connectivity metric and considered the alpha (8-12Hz) spectral band.

We tested two settings. Firstly, a train-test scheme where connectivity matrices are obtained for the two datasets separately and the weight matrices of the test set (dataset-1, MCI) are denoised according to the graph metrics of the training dataset (dataset-2). Secondly, a within-subject denoising setting where the connectivity matrix of the first 50 epochs of a subject (from dataset-2, FAD) is considered as the training weight matrix $\mathbf{W}_{tr}$, and each of the other

TABLE 1
Denoising of noisy EEG networks of 28 channels. For each of the 32 test subjects (19 controls - 13 patients from dataset-1) we denoised each trial of the subject's weight matrix based on the graph metrics of the mean weight matrices of the two groups from the training set (dataset-1) and its own weight matrix. We show the average MSE over all subjects for the three denoising schemes and the original noisy network. Error is computed is against $\mathbf{W}_{tr}$.

Same Group	Opposite Group	Self	Noisy original
5.31 ± 0.79	5.33 ± 0.88	4.51 ± 0.66	6.26 ± 1.21

matrices is a test matrix $\mathbf{W}_i$ to be denoised. Here, we used the transitivity and degree as graph metrics combining both the global and local properties of brain activity.

In Table I we show the results of the denoising algorithm when using as target values the graph metrics of a) the same subject, b) the same group (patient/control), c) the opposite group. The mean square error (MSE) is computed against the training network $\mathbf{W}_{tr}$. Differences between subjects were significant under a unpaired ttest, $p < 0.01$, for all pairwise comparisons except between the same group and opposite group. In Figure 5 we show an example of denoised networks for one trial of a subject.

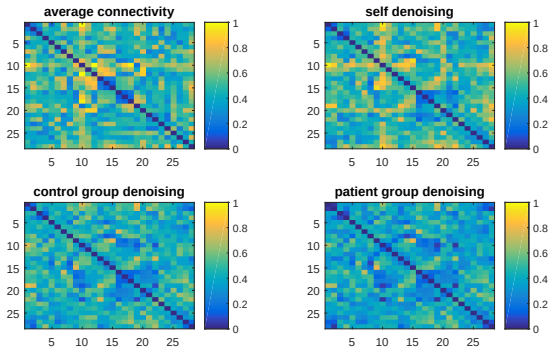

Fig. 5. Example of obtained denoised networks for one a trial of a single subject. The network denoised based on the subject specific graph metrics retained the basic characteristics of the true network.

4.1.3 Macaque connectivity

In order to further demonstrate the potential applicability of the algorithm, we obtained the connectivity matrices from the brains of two Macaques [40] with similar values for the transitivity metric. In Figure 6 we show the result of the denoising where the first Macaque's (M1) brain was used to estimate the transitivity and drive the denoising for the second Macaque (M2).

4.2 Completion

4.2.1 Synthetic Networks

Here we show the results of the network completion Algorithm 2 for an increasing number of missing entries and number of nodes. Each separate case considers the average of 50 noise matrix realisations.Here we deal with a random network with $w_{ij} \sim \mathcal{U}[0, 1]$. For each case, we optimise three graph metrics (transitivity, degree, clustering coefficient) and show the error reduction of the completion procedure.

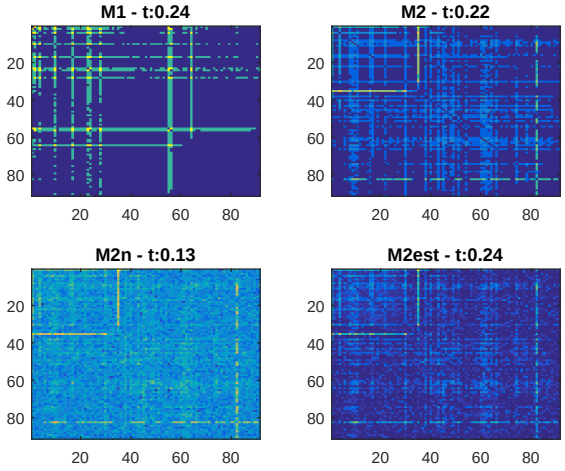

Fig. 6. Example of obtained denoised networks when using M1's data to estimate the transitivity and apply that to the network of M2. It is observed that the denoised network is close to the true network even though the transitivity was set to that of M1.

The missing values are initialised to 0.5 since this is the mean of the uniform $\mathcal{U}[0, 1]$ distribution. This initialisation produces the smallest distance to the true network from all possible initialisations. Error reduction is calculated the same way as in Eq. 24 with $\mathbf{W}_e$ being the initialised network. Results are shown in Figure 7.

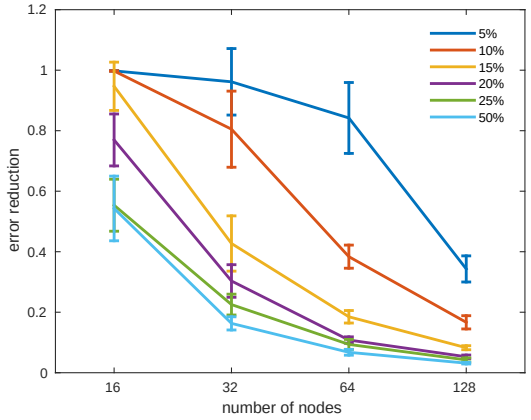

Fig. 7. Error reduction of the completion procedure for different number of nodes and percentage of missing entries. We use here three graph metrics (degree, transitivity, clustering).

4.2.2 Real Networks

4.2.2.1 Known metrics: We applied the completion algorithm on the following datasets. 1) USAir: a 330 node network of US air transportation, where the weight of a link is the frequency of flights between two airports [41]. 2) Baywet: a 128 node network which contains the carbon exchanges in the cypress wetlands of south Florida during the wet season [42]. The weights indicate the feeding levels between taxons. 3) Celegans: the neural network of the worm C. elegans. Nodes indicate neurons and the edges are weighted by the number of synapses between the neurons

TABLE 2
Further error reduction of Algorithm 4 compared to only denoising the separate networks. In this case we have mixed a modular with a scale-free network. With this figure we illustrate that the optimisation problem of Eq. 6 results in reducing the error between the estimates and the true networks as would happen from problem in Eq. 8.

Nodes	16	32	64	128
Reduction	0.35 ± 0.07	0.30 ± 0.24	0.24 ± 0.30	0.22 ± 0.22

[43]. In Figure 8 we show the results of the completion algorithm 2 for all three networks and different graph metrics.

4.2.2.2 Uncertain metrics: In this section we demonstrate the performance when the network metrics come from different datasets. This showcases a realistic scenario when a similar type of network is utilised in order to complete the missing values. In Figure 9 we show the results of the completion (a) on the Baywet dataset with the metric obtained in the Baydry dataset which contains the carbon exchanges in the dry season [40] and (b) on two enzyme network with the metrics estimated from the g355 enzyme and the completion performed on the g504 enzyme [40].

4.3 Decomposition

4.3.1 Synthetic Networks

In this section we show example results of the decomposition scheme by considering by mixing (as their sum) a modular and scale-free network. The modular network consists of 8 modules. For the modular network we use the modularity and for the scale free network we use the transitivity as graph metrics to optimise. In Table II we show the average error reduction of the two networks of Algorithm 4 as compared with only denoising the two networks separately. In this cases error reduction for a single network is defined as:

$$er = 1 - \frac{||\mathbf{W}_{dec} - \mathbf{W}||_{\mathrm{F}}}{||\mathbf{W}_{den} - \mathbf{W}||_{\mathrm{F}}} \tag{24}$$

For all cases the penalty parameter λ is adjusted such that the reconstruction error is reduced w.r.t. iterations of the alternating minimisation procedure.

4.3.2 Airline data

The dataset is a binary network that contains the networks for 37 airlines and 1000 airports [44]. We converted each airline's binary network to a weighted network by adjusting any existing edge between two airports to the total number of edges for all airlines. The mixed network consists of two networks of different airlines mixed together (Lufthansa, Ryanair) for a subset of 50 nodes (airports). In Figure 10 we show the decomposition result by using only global graph metrics (transitivity and global clustering coefficient).

5 Discussion

The optimisation schemes described in this work enable the adjustment of a network's weight matrix to fulfil specific properties. The utility of the denoising scheme (Section 3.1) was evaluated on a number of cases including real-world data. It has to be pointed out that for convex graph metrics, the denoising scheme is guaranteed to converge to a network that is closer to the true underlying network than the noisy observed network. In Figure 1 it is observed that considering the degrees of the nodes overshadows any other global metric's performance. For non convex graph metrics there is no guarantee but as shown in Figures 1 and 2, the estimated network is a better estimate of the underlying network than the original noisy version, even for increasing noise and different network types. The only exceptions to this can be observed for the clustering metrics (transitivity, clustering coefficient) of the modular network in Figure 3. This can be explained by noting that in our example most of the weights (90%) are clustered in modules while the denoising algorithm operates on all the weights. Constraining the weight updates only in the modules alleviates that problem. The utility of this scheme is also displayed in Figure 4 where an increase in the number of nodes does not affect the performance. We point out that although the error reduction increases as the number of nodes increases, the error actually increases in absolute terms.

We also tested the efficacy of the scheme in a real world EEG connectivity dataset of Alzheimers patients and control subjects. When performed in a within subject fashion, and splitting each subject's data into a train set to obtain estimates of the graph metrics, and a test set to apply the denoising algorithm, we successfully reduced the variability of the network regarding background EEG activity. More importantly, though, in a leave-subject-out procedure; using other subject's graph metrics as prior knowledge the algorithm was able to decrease the noise of the network, albeit not as much as in the within subject paradigm as expected. This has important implications in the EEG and related fields (e.g. BCI, fMRI, DTI) where subject independent paradigms are necessary to obtain practically usable and consistent performance [45]. This is supported by the Macaque example where the transitivity metric of one Macaque was used to drive the denoising procedure for the connectivity matrix of another Macaque, see Figure 6. Furthermore, such an approach can be useful in any application that can obtain prior knowledge of the structure of the network under consideration. Our work extends prior work in the network denoising field [23]–[25] by providing theoretical proofs and empirical evidence that it is viable for a variety of network types. We also provide the strong proof that the degree of a network, resulting in a convex cost function, is very important in network estimation.

Weighted network completion has not been attempted in the literature and here we employed graph metrics as the driving force behind estimating the missing weights. For modest sizes of missing entries we showed that there is significant benefit of using the completion Algorithm 2 up to 128 node networks and using three graph metrics. Similarly, for real networks, we show that the knowledge of graph metrics can aid in completing the network. More importantly, there is a benefit from the knowledge of global metrics as seen in Figure 8. For the cases that the metrics are not completely known we show the efficacy of the completion algorithm in two cases where we use similar datasets to drive the completion procedure, see Figure 9.

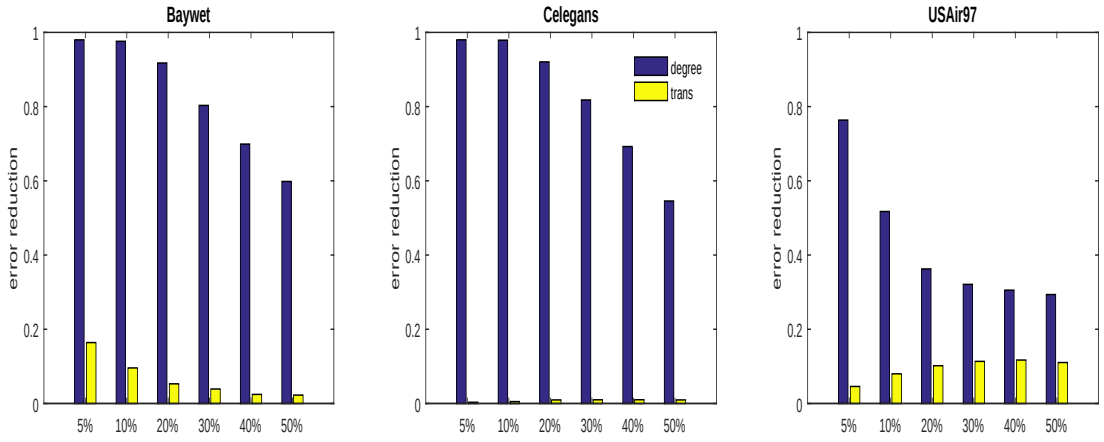

Fig. 8. Error reduction of the completion procedure for the three networks in terms of the percentage of missing entries. The estimated network was closer to the true network for both global (transitivity) and local (degree) metrics.

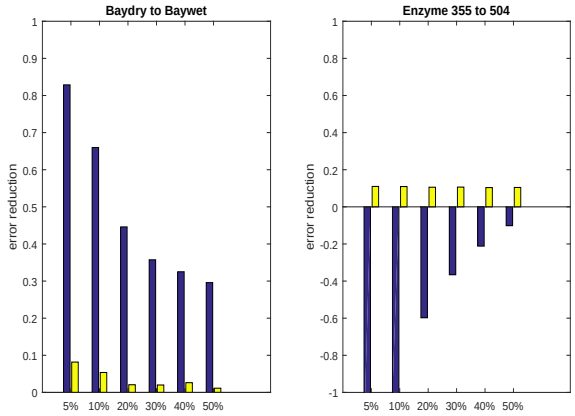

Fig. 9. Error reduction of the completion procedure when using network metrics from different sources. For the Baywet dataset, the estimated network was closer to the true network for both global (transitivity) and local (degree) metrics. On the other hand, for the enzyme networks only the global metric resulted in a decrease in the error due to the large individual differences.

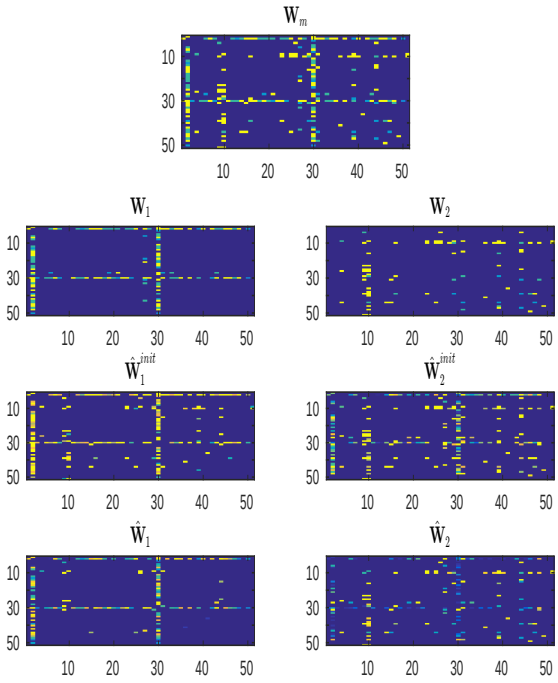

Fig. 10. Decomposition of a mixed airline network $\mathbf{W}_f$ of 50 nodes into estimates $\hat{\mathbf{W}}_1$ and $\hat{\mathbf{W}}_2$. The true networks are shown as $\mathbf{W}_1$ and $\mathbf{W}_2$. The initial estimates of the denoising algorithm for the two networks are shown as $\mathbf{W}_1^{init}$ and $\mathbf{W}_2^{init}$. In this case we have used only global metrics (transitivity and clustering coefficient).

When utilising graph metrics in the network decomposition scheme (Section 3.3) the first observation is that the error reduction is greater by using information from both networks that combine to produce the mixed network. As shown in Table II the resulting estimates from the alternating minimisation procedure of Algorithm 4 confirm the intuition behind that methodology. It has to be noted that this is expected since more information is used as compared to only denoising. Namely, that the constraints employed in the optimisation problems of (6) and (7) can result in the behaviour of optimisation problem (8). Also in Table II, we show the error reduction as a function of the number of nodes. The choice of global graph metrics, modularity and transitivity, further demonstrates the utility of the methodology in setting where local information would be difficult to obtain.

The decomposition algorithm was tested on real data on mixed airline networks for both global and local only metrics. Assuming prior information on the structure of the networks, the algorithm was able to produce reliable estimates for the underlying networks, see Figure 10. Network decomposition for mixed networks has been rarely attempted in the literature. Our work provides a new formulation that takes account prior knowledge and the use of the reconstruction error in the estimation process. This is in contrast with [24], [31] where only factor graphs are considered.

The choice of which graph measure should be used depends on the application and on which may be available.

Local metrics assume knowledge of individual nodes' properties as is typical for e.g EEG applications (Figure 5) but may not be the case for the airport data (Figure 10). There are two limitations of this study that will be addressed in future work. Computational complexity and scaling the efficacy to large networks. Although many real world networks (e.g. EEG, Social Networks, Weather Networks, Airline Networks) are of the size that we consider in this work ($100s$ of nodes) there exist networks that consist of very large number of nodes ($>> 1000s$ of nodes). The calculation of the derivatives increases in the order of $\mathcal{O}(n^2)$ making the computation slow and inefficient. As also discussed in [46], this is an open issue in graph based metrics. For large computational operations on graphs, polynomial approximations have been proposed [47]. Similarly, as a network increases in size, the number of graph metrics should be increased in order to produce comparable performance to the medium sized networks considered here. This is because the number of unknown parameters to estimate increases therefore more graph metrics are necessary. In order to complement the popular graph metrics that we used in this work, we are considering other potential candidates such as ones based on graph spectral methods, and statistical graph analysis.

6 Conclusions

In this work we developed three mathematical optimisation frameworks that were utilised in network denoising, decomposition and completion. The basis of the methodology lies in adjusting a network's weights to conform with known graph measure estimates. We derived expressions for the derivatives of popular graph metrics and designed algorithms that use those derivatives in gradient descent schemes. We tested our proposed methods in toy examples as well as real world datasets.

The work performed here has the following implications for network estimation. Firstly, we showed that the use of graph metrics for network denoising reliably reduces the noise in an observed network for both convex and non-convex graph metrics. Also, by combining multiple graph metrics, further reduction is ensued. Depending on the type of network, some metrics may be more appropriate than others. Modularity works well for modular networks while degree seems to perform well for both random and scale free networks. For network decomposition, the use of global information as prior knowledge was sufficient to separate the underlying networks from their mixture. Such a framework can be the basis for constrained matrix or tensor decomposition of dynamic networks or multilayer networks. Finally, we provide a new weighted network completion paradigm that can complement existing matrix completion algorithms.

Other applications of our methodology can be weighted network reconstruction; the field that designs networks from scratch fullfilling specific criteria (e.g. a specific value for transitivity). The design of such network from scratch can be performed by transversing the level sets of a graph measure through the graph measure derivatives. Link prediction can also be incorporated by considering not only the weight similarities between different nodes but also the similarity between their derivatives.

Acknowledgment

We would like to thank Dr Mario Parra Rodriguez (Heriot-Watt University) for making the EEGs available to us.

References

[1] B. Bollobas, *Graph theory: an introductory course*. Springer Science & Business Media, 2012.
[2] J. L. Gross and J. Yellen, *Graph theory and its applications*. CRC press, 2005.
[3] N. Deo, *Graph theory with applications to engineering and computer science*. Courier Dover Publications, 2016.
[4] A.-L. Barabási, "Network science: Luck or reason," *Nature*, vol. 489, no. 7417, pp. 507–508, 2012.
[5] N. M. Tichy, M. L. Tushman, and C. Fombrun, "Social network analysis for organizations," *Academy of management review*, vol. 4, no. 4, pp. 507–519, 1979.
[6] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
[7] A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani, "The architecture of complex weighted networks." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 11, pp. 3747–3752, 2004.
[8] M. Rubinov and O. Sporns, "Complex network measures of brain connectivity," *NeuroImage*, vol. 52, pp. 1059–1069, 2010.
[9] A. Laita, J. S. Kotiaho, and M. Mönkkönen, "Graph-theoretic connectivity measures: What do they tell us about connectivity?" *Landscape Ecology*, vol. 26, no. 7, pp. 951–967, 2011.
[10] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American Society for Information Science and Technology*, vol. 58, no. May 2007, pp. 1019–1031, 2007.
[11] D. S. Goldberg and F. P. Roth, "Assessing experimentally derived interactions in a small world." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 8, pp. 4372–4376, 2003.
[12] L. Lu and T. Zhou, "Link prediction in complex networks: a survey," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150–1170, 2010.
[13] M. Kim and J. Leskovec, "The Network Completion Problem: Inferring Missing Nodes and Edges in Networks," *SIAM International Conference on Data Mining*, pp. 47–58, 2011.
[14] S. Hanneke and E. P. Xing, "Network Completion and Survey Sampling," *Aistats*, vol. 5, pp. 209–215, 2009.
[15] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.
[16] P. Symeonidis, N. Iakovidou, N. Mantas, and Y. Manolopoulos, "From biological to social networks: Link prediction based on multi-way spectral clustering," *Data & Knowledge Engineering*, vol. 87, pp. 226–242, 2013.
[17] R. Mastrandrea, T. Squartini, G. Fagiolo, and D. Garlaschelli, "Enhanced reconstruction of weighted networks from strengths and degrees," *New Journal of Physics*, vol. 16, 2014.
[18] T. Squartini and D. Garlaschelli, "Analytical maximum-likelihood method to detect patterns in real networks," *New Journal of Physics*, vol. 13, 2011.
[19] K. Bleakley, G. Biau, and J.-P. Vert, "Supervised reconstruction of biological networks with local models," *Bioinformatics*, vol. 23, no. 13, pp. 57–65, 2007.
[20] M. Filosi, R. Visintainer, S. Riccadonna, G. Jurman, and C. Furlanello, "Stability indicators in network reconstruction," *PLoS ONE*, vol. 9, no. 2, 2014.
[21] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '10*, vol. 5, no. 4, pp. 1019–1028, 2010.
[22] S. Myers and J. Leskovec, "On the convexity of latent social network inference," in *Advances in Neural Information Processing Systems*, 2010, pp. 1741–1749.

[23] M. Aghagolzadeh, M. Al-Qizwini, and H. Radha, "Denoising of network graphs using topology diffusion," *Conference Record - Asilomar Conference on Signals, Systems and Computers*, vol. 2015-April, no. 1, pp. 728–732, 2015.
[24] Q. D. Morris and B. J. Frey, "Denoising and Untangling Graphs Using Degree Priors," *Advances in Neural Information Processing Systems 16*, pp. 385–392, 2004.
[25] H. Gao, X. Wang, J. Tang, and H. Liu, "Network denoising in social media," *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining - ASONAM '13*, pp. 564–571, 2013.
[26] G. Caldarelli, A. Chessa, F. Pammolli, A. Gabrielli, and M. Puliga, "Reconstructing a credit network," *Nature Physics*, vol. 9, no. 3, pp. 125–126, 2013.
[27] O. Sporns and R. F. Betzel, "Modular Brain Networks," *Annual Review Psychology*, vol. 67, pp. 613–640, 2016.
[28] B. Huang and T. Jebara, "Exact graph structure estimation with degree priors," *8th International Conference on Machine Learning and Applications, ICMLA 2009*, pp. 111–118, 2009.
[29] R. H. Keshavan, A. Montanari, and S. Oh, "Matrix completion from a few entries," *IEEE Transactions on Information Theory*, vol. 56, no. 6, pp. 2980–2998, 2010.
[30] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst, "Matrix Completion on Graphs," *arXiv*, p. 10, 2014. [Online]. Available: http://arxiv.org/abs/1408.1717
[31] C. Nash-Williams, "Decomposition of finite graphs into forests," *Journal of the London Mathematical Society*, vol. 39, no. 12, pp. 157–166, 1964.
[32] M. Barthélemy, A. Barrat, R. Pastor-Satorras, and A. Vespignani, "Characterization and modeling of weighted networks," *Physica A: Statistical Mechanics and its Applications*, vol. 346, no. 1-2 SPEC. ISS., pp. 34–43, 2005.
[33] Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.
[34] G. Fagiolo, "Clustering in Complex Directed Networks," *Reason*, vol. 162, no. August, pp. 139– 162, 2004.
[35] J. P. Onnela, J. Saramäki, J. Kertész, and K. Kaski, "Intensity and coherence of motifs in weighted complex networks," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 71, no. 6, pp. 1–4, 2005.
[36] T. Opsahl and P. Panzarasa, "Clustering in weighted networks," *Social Networks*, vol. 31, no. 2, pp. 155–163, 2009.
[37] Y.-T. Chang and D. Pantazis, "Modularity Gradients: Measuring the contribution of edges to the community Structure of a brain network," *IEEE 10th International Symposium on Biomedical Imaging*, pp. 536–539, 2013.
[38] B. J. Prettejohn, M. J. Berryman, and M. D. McDonnell, "Methods for generating complex networks with selected structural properties for simulations: a review and tutorial for neuroscientists." *Frontiers in computational neuroscience*, vol. 5, no. March, p. 11, 2011.
[39] M. Pietto, M. A. Parra, T. N., F. F., G. A.M., B. J., R. P., M. F., L. F. I. A., and B. S., "Behavioral and Electrophysiological Correlates of Memory Binding Deficits in Patients at Different Risk Levels for Alzheimers Disease," *Journal of Alzheimer's Disease*, vol. 53, pp. 1325–1340, 2016.
[40] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. [Online]. Available: http://networkrepository.com
[41] "Pajek dataset; 2006. available: http://vlado.fmf.uni-lj.si/pub/networks/data/."
[42] "The koblenz network collection; 2015. available: http://konect.uni-koblenz.de/."
[43] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks." *Nature*, vol. 393, no. 6684, pp. 440–2, 1998.
[44] A. Cardillo, J. Gómez-Gardenes, M. Zanin, M. Romance, D. Papo, F. del Pozo, and S. Boccaletti, "Emergence of network features from multiplexity," *arXiv preprint arXiv:1212.2153*, 2012.
[45] L. Spyrou, Y. Blokland, J. Farquhar, and J. Bruhn, "Optimal multitrial prediction combination and subject-specific adaptation for minimal training brain switch designs," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. PP, no. 99, 2015.
[46] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.
[47] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129 – 150, 2011.

L oukianos Spyrou is a Research Associate at the Institute for Digital Communications, University of Edinburgh's School of Engineering. He is currently working on the MASNET project with Prof. John Thompson. His primary research interests are in machine learning and signal processing methodologies with focus on biomedical applications. Loukianos received the M.Eng. degrees from the University of York, Department of Electronics in 2004, the M.Sc. degree from the King's College London in 2005. His PhD was awarded in 2009 from Cardiff University. He has been working as a postdoctoral researcher since 2012 with previous posts in Radboud University and in the University of Surrey.

J avier Escudero (S07M10) received the MEng and PhD degrees in telecommunications engineering from the University of Valladolid, Spain, in 2005 and 2010, respectively. Afterwards, he held a post-doctoral position at Plymouth University, UK, until 2013. He is currently a tenured faculty member (Chancellor's Fellow) at the School of Engineering of the University of Edinburgh, UK, where he leads a research group in biomedical signal processing with particular interests in non-linear analysis, network theory, and multi-way decompositions. He is author of over 45 scientific articles. Dr Escudero received the Third Prize of the EMBS Student Paper Competition in 2007 and the award to the best PhD thesis in healthcare technologies by the Spanish Organization of Telecommunications Engineers in 2010. In 2016, he was elected member of the Young Academy of Scotland. He is president of the society of Spanish Researchers in the United Kingdom during the 2018/19 term.

APPENDIX A

Theorem 1. *Let $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^n$ and $f(x) = M$ with $f : \mathbb{R}^n \to R$. Suppose $\boldsymbol{z} = \boldsymbol{x} + \boldsymbol{y}$. If the cost function $c(\boldsymbol{z}) = (f(\boldsymbol{z}) - M)^2$ is convex, then gradient descent with iteration index $k \in (0...K)$:*

$$\boldsymbol{z}^{k+1} = \boldsymbol{z}^k - \lambda \nabla c(\boldsymbol{z}^k) \tag{1}$$

will converge to a point $\boldsymbol{z}^K$ such that:

$$||\boldsymbol{x} - \boldsymbol{z}^K||_2 \leq ||\boldsymbol{x} - \boldsymbol{z}||_2 \tag{2}$$

for any $\boldsymbol{z}$ and small enough λ.

Proof. The vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$ denote the vectorised versions of the adjacency matrices $\mathbf{W}$. This theorem describes the situation that a network ($\mathbf{x}$) is corrupted by noise ($\mathbf{y}$) resulting in a noisy network ($\mathbf{z} = \mathbf{x} + \mathbf{y}$). The purpose of this proof is to show that gradient descent with convex $c(.)$ will converge to $\mathbf{z}^K$ which is a better, in terms of distance, estimate of $\mathbf{x}$ than $\mathbf{z}$. The proof does not try to show that gradient descent on convex functions achieves a global minimum; instead that the minimum achieved is a better estimate of the network than the original noisy network $\mathbf{z}$. Note that for graph metrics in general, there are infinite solutions for a matrix $\mathbf{W}$ that achieves a minumum point.

For strictly convex functions, the proof is trivial since there is a unique minimum which corresponds to $\mathbf{x}$ only and gradient descent will converge to that value.

For both convex and strictly convex functions the proof is as follows. The sublevel sets, $L_s(c) = \{\mathbf{z} \in \mathcal{R}^n \mid c(\mathbf{z}) < s\}$ of a convex function are convex sets. Furthermore, $L_a(c) \subseteq L_b(c)$ for any $a \leq b$. For any point $\mathbf{z}^k$ with $c(\mathbf{z}^k) = L_k$, the negative gradient $-\nabla c(\mathbf{z}^k)$ forms a right angle with the level set at L_k (by definition). Since the level sets are convex sets, the gradient update $\mathbf{z}^k - \lambda \nabla c(\mathbf{z}^k)$ leads to a point $\mathbf{z}^{k+1}$ such that the angle between $\mathbf{z}^{k+1}\mathbf{z}^k$ and $\mathbf{z}^k\mathbf{z}^*$ is acute in the triangle $(\mathbf{z}^k)(\mathbf{z}^{k+1})(\mathbf{z}^*)$ for any $\mathbf{z}^* \in L_k(c)$ (see Figure **??**) and small enough step size. Since the optimum point $\mathbf{x}$ is also contained in $L_k(c)$ and the relation between the distances $||\mathbf{z}^* - \mathbf{z}^{k+1}|| < ||\mathbf{z}^* - \mathbf{z}^k||$, this implies that $||\mathbf{x} - \mathbf{z}^{k+1}|| < ||\mathbf{x} - \mathbf{z}^k||$ for any $\mathbf{z}$ and $\mathbf{k}$ provided that the step size is small enough. Small enough in the sense that the gradient step must not cross the level set of $\mathbf{z}^*$. □

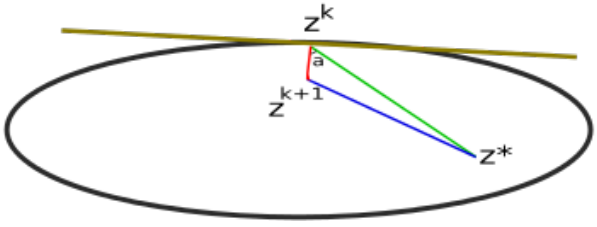

Fig. 1. Triangle formed between z^k, z^{k+1} and z^*. z^* is any point inside the sublevel set at $f(z^k) = K_k$. For such a triangle the distance $||z^* - z^{k+1}||$ is always smaller than $||z^* - z^k||$. The closed curve denotes the boundary of the level set at k.

APPENDIX B

B.1 Derivatives of scalar functions of matrices

Differentiating a scalar function $f(\mathbf{W})$ w.r.t. a matrix $\mathbf{W}$, $\frac{\partial f}{\partial \mathbf{W}}$, is essentially a collection of derivatives w.r.t. the separate matrix elements placed at the corresponding indices, i.e.:

$$\frac{\partial f}{\partial \mathbf{W}} = \begin{bmatrix} \frac{\partial f}{\partial w_{11}} & \frac{\partial f}{\partial w_{12}} & \cdots & \frac{\partial f}{\partial w_{1n}} \\ \frac{\partial f}{\partial w_{21}} & \frac{\partial f}{\partial w_{22}} & \cdots & \frac{\partial f}{\partial w_{2n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial w_{n1}} & \frac{\partial f}{\partial w_{n2}} & \cdots & \frac{\partial f}{\partial w_{nn}} \end{bmatrix} \tag{3}$$

Expressing the derivatives in matrix form allows easy and scalable formulation of the derivatives irrespective of the number of entries. If the derivative of a specific element of $\mathbf{W}$, indexed by (i, j), is required to be processed separately, this can be performed by selecting the same index of the derivative matrix. For example $\{\frac{df}{d\mathbf{W}}\}_{ij} = \frac{df}{dw_{ij}}$.

In the case of undirected networks, where the weight matrices are symmetric, the following adjustment needs to be made to ensure that the derivatives are themselves symmetric:

$$\frac{df}{d\mathbf{W}} = \frac{\partial f}{\partial \mathbf{W}} + \left(\frac{\partial f}{\partial \mathbf{W}}\right)^T - diag\left(\frac{\partial f}{\partial \mathbf{W}}\right) \tag{4}$$

The formulations in the text are in terms of the partial derivatives for simplicity.

B.2 Modularity derivative

The modularity is written as:

$$M = \frac{1}{l^w} \sum_{ij} \left(w_{ij} - \frac{k_i^w k_j^w}{l^w}\right) \delta_{ij} \tag{5}$$

where $\delta_{ij} = 1$ whenever nodes i and j belong to the same module and zero otherwise. The term $\frac{1}{l^w} \sum_{ij}(w_{ij}\delta_{ij})$ can be written as:

$$m_1 = \frac{1}{l^w} \sum_{ij}(w_{ij}\delta_{ij}) = \frac{tr\{\mathbf{W}\boldsymbol{\Delta}^\mathbf{T}\}}{tr\{\mathbf{W}\mathbf{O_n}\}} = \frac{\theta}{l^w} \tag{6}$$

where $\boldsymbol{\Delta}$ is the matrix that contains the δ_{ij}. The gradient is given by:

$$\frac{\partial m_1}{\partial \mathbf{W}} = \frac{l^w \boldsymbol{\Delta} - \theta \mathbf{O_n^T}}{(l^w)^2} \tag{7}$$

The other term can be written as:

$$m_2 = \frac{1}{(l^w)^2} \sum_{ij} k_i^w k_j^w \delta_{ij} = \frac{1}{(l^w)^2} \sum_{ijkl} w_{ik} w_{jl} \delta_{ij} = \tag{8}$$

$$\frac{1}{(l^w)^2} \sum_{r=1}^{n} tr\{\mathbf{W}^\mathbf{T}\mathbf{C_r}\mathbf{W}\boldsymbol{\Delta}^\mathbf{T}\} = \frac{1}{(l^w)^2} \sum_{r=1}^{n} \xi_r \tag{9}$$

where $\mathbf{C_r}$ is a circular shift matrix that shifts down the rows of the matrix on the right by $r - 1$.

APPENDIX C

Theorem 2. *For any weighted network with a graph measure of the type $f(\boldsymbol{W}) = tr\{\boldsymbol{WA}\}$ the function $g(\boldsymbol{W}) = (f(\boldsymbol{W}) - K)^2$ is convex for any matrix $\boldsymbol{A}$.*

Proof. In order to show that $g(\mathbf{W})$ is convex it suffices to show that the Hessian of g is positive semidefinite. We will use differential notation **[?]** to calculate the Hessian which is defined as:

$$\{\mathbf{H}g\}_{ij} \equiv \frac{\partial^2 g}{\partial x_i \partial x_j} \tag{10}$$

where x_i is an element of the vectorized weight matrix $\mathbf{W}$. The first differential of $g(\mathbf{W})$ is:

$$dg(\mathbf{W}) = 2\left(tr\{\mathbf{WA}\} - K\right) tr\{d\mathbf{WA}\} \tag{11}$$

The second differential is:

$$d^2 g(\mathbf{W}) = 2tr\{d\mathbf{WA}\}tr\{d\mathbf{WA}\} \tag{12}$$

Using the relation between the trace and the vec operator, i.e. $tr\{\mathbf{A}^T\mathbf{B}\} = vec(\mathbf{A})^T vec(\mathbf{B})$ and the circular property of the trace, i.e. $tr\{d\mathbf{WA}\} = tr\{\mathbf{A}d\mathbf{W}\}$:

$$d^2 g(\mathbf{W}) = 2vec(\mathbf{dW}^T)^T vec(\mathbf{A}) vec(\mathbf{A}^T)^T vec(\mathbf{dW}) \tag{13}$$

$$= 2vec(\mathbf{dW})^T \mathbf{K}_{nn} vec(\mathbf{A}) vec(\mathbf{A}^T)^T vec(\mathbf{dW}) \tag{14}$$

$$= vec(\mathbf{dW})^T 2vec(\mathbf{A}^T) vec(\mathbf{A}^T)^T vec(\mathbf{dW}) \tag{15}$$

where $\mathbf{K}_{nn}$ is the commutation matrix satisfying $vec(\mathbf{X}^T) = \mathbf{K}_{nn} vec(\mathbf{X})$. Note that we have 'commuted' $\mathbf{K}_{nn}$ from $vec(\mathbf{dW}^T)$ to $vec(\mathbf{A}^T)$. The second differential was brought to the form $d^2 g = vec(d\mathbf{W})^T \mathbf{Z} vec(d\mathbf{W})$ which means that the Hessian is **[?]**:

$$\mathbf{H}g = \frac{1}{2}(\mathbf{Z} + \mathbf{Z}^T) \tag{16}$$

The matrix $\mathbf{Z}$ is of the type $\mathbf{Z} = \mathbf{cc}^T$. Since $\mathbf{c}$ is a vector the product $\mathbf{cc}^T$ is rank-one producing a single nonzero positive eigenvalue. Therefore $\mathbf{Z}$ is positive semi-definite and hence the Hessian is positive semi-definite. □